PATENT
5150-53800

"EXPRESS MAIL" MAILING LABEL
NUMBER EL893746806US
DATE OF DEPOSIT NOVEMBER 28, 2001
I HEREBY CERTIFY THAT THIS PAPER OR FEE IS BEING DEPOSITED WITH THE UNITED STATES POSTAL SERVICE "EXPRESS MAIL POST OFFICE TO ADDRESSEE" SERVICE UNDER 37 C.F.R. § 1.10 ON THE DATE INDICATED ABOVE AND IS ADDRESSED TO THE COMMISSIONER OF PATENTS AND TRADEMARKS, ARLINGTON, VA 22202

Derrick Brown

MOTION CONTROL SYSTEM AND METHOD WHICH INCLUDES IMPROVED PULSE PLACEMENT FOR SMOOTHER OPERATION

By:

Joseph E. Peck
Rodger Schorr
Neil Feiereisel

Attorney Docket No.: 5150-53800

Jeffrey C. Hood/RPH
Conley, Rose & Tayon, P.C.
P.O. Box 398
Austin, Texas 78767-0398
Ph: (512) 476-1400

Title: Motion Control System and Method Which Includes Improved Pulse Placement for Smoother Operation

Inventors: Joseph E. Peck, Rodger Schorr, Neil Feiereisel

FIELD OF THE INVENTION

The present invention relates generally to motion control. More particularly, the present invention relates to a system wherein a motion control system uses pulses to instruct a motion device to move an object.

DESCRIPTION OF THE RELATED ART

Motion control is a broad term that may be defined as the precise control of anything that moves. A motion system typically comprises five major components: 1) the moving mechanical device; 2) the motor (servo or stepper motor) with feedback and motion I/O; 3) the motor drive unit; 4) the intelligent controller; and 5) the programming/interface software. Scientists and engineers typically use servo and stepper motors for position and velocity control in a variety of electro-mechanical configurations.

In particular, stepper motor systems typically include a controller, a power drive, and a stepper motor. The controller is able to generate step pulses to command the drive to move the motor (and therefore the object that is desired to be moved) an incremental movement often called a "step." The drive accepts these pulses and generates the high currents and voltages necessary to move the motor. The frequency of the step pulses controls velocity, the rate of change controls acceleration, and the total number of pulses controls the position.

Prior motion control systems have used proprietary control hardware to control the motion system. These proprietary systems have suffered from high cost and limited flexibility. More recently, computer systems are being used in motion control systems.

The computer system may serve as the operator interface or human machine interface (HMI) as well as the local control host in the remote system controller platform. The use of personal computers in motion control is widely accepted and growing at a significant pace. While many motion control solutions today still use standalone distributed motion control and closed architecture systems, computer-based motion solutions provide added flexibility and the potential for lower system cost.

In computer-based stepper motion control systems, it is common to segment the total motion into short time intervals. During each interval (i.e., each iteration of the loop), the controller decides where the motor should be at the end of the interval. The controller then outputs the number of step pulses equal to the difference between the target position and the current position. It is also common practice to evenly distribute the required number of pulses across the loop period. However, this even distribution can result in significant short-term velocity and position error.

For example, for a loop period of 10 clocks and a step rate of 7 clocks, steps may be generated as follows according to the prior art method:

Period	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Target Position	1.4	2.9	4.3	5.7	7.1	8.6	10
Steps to generate	1	1	2	1	2	1	2
Actual step rate	10	10	5	10	5	10	5

Because the motion control systems of the prior art use integer values for steps, the instantaneous step rate (i.e., the step rate per period) is either 5 or 10 clocks even though the average step rate is 7 clocks. Figure 4A illustrates a typical graph (of velocity versus time) having significant short-term error according to the prior art motion control system and method.

Prior art motion control systems also suffer from quantization errors, primarily because they can only output step rates that are an integer number of clock cycles. If a digital motion control system can only output step rates that are an integer number of clocks, for example, then it can only choose a step rate of 2 or 3 clocks (rather than an ideal

2.4 clocks, for example). If it uses a step rate of 2 clocks, then the pulse train will end early in the period and leave dead time that creates jumps in velocity. If it uses a step rate of 3 clocks, the pulse train will not finish by the end of the period and will run into the next period.

Therefore, an improved system and method is desired for motion control using improved pulse placement for smoother operation.

SUMMARY OF THE INVENTION

One embodiment of the present invention includes a motion control system and method which provides improved pulse placement for smoother operation of a motion device such as a stepper motor. Although prior art implementations typically do generate the correct number of steps at the correct average velocity, they do so at the expense of short-term error. At both high and low velocities, the motion control system and method as described herein will typically result in smoother operation as well as achieve positional accuracy through accurate pulse placement.

The motion device (e.g., a stepper motor) is operable to move an object. The motion device is coupled to a motion control system which may include a computer system and a motion controller. The motion control system may include a processor and a memory medium, wherein the memory medium stores a motion control software program which is executable by the processor. A power drive may be coupled between the motion device and the motion control system. The power drive may be operable to receive the pulses from the motion control system, translate the pulses into power signals, and send the power signals to the motion device.

In one embodiment, to achieve smoother operation, the motion control system and method may place the step pulses more accurately within the loop period. By using a delay time, the pulse train may be shifted to an arbitrary location within the loop period rather than evenly distributed throughout the loop period as in the prior art implementations.

In one embodiment, the motion control system and method may correct for quantization errors in the step generation due to digital clock limits. In one embodiment, the motion control system may generate a pulse train at the slower rate (3 clocks in this example) and correct for the "borrowed time" in the next loop iteration. Instead of assuming that each loop period is constant, according to one embodiment of the motion control system and method, an autocorrecting algorithm removes the borrowed time from its calculations and therefore allows the step generation to catch up at appropriate intervals.

In one embodiment, a further improvement may be made to improve the accuracy of the motion control system and method. By allowing the step rate to change at a programmable point in the middle of the loop iteration (i.e., the series of loop periods), the step pulse generator may allow the steps generated to consume only the loop period and thus eliminate the borrowing of time from future loop periods.

BRIEF DESCRIPTION OF THE DRAWINGS

A better understanding of the present invention can be obtained when the following detailed description of the preferred embodiment is considered in conjunction with the following drawings, in which:

Figure 1 illustrates an example of a system for motion control and measurement including an object being scanned according to one embodiment;

Figure 2 illustrates an example of a motion control system according to one embodiment;

Figure 2A illustrates an example of a motion control system having a motion control interface device and a data acquisition device comprised within a computer system according to one embodiment;

Figure 2B illustrates an example of a motion control system having a motion control interface device and an image acquisition device comprised within a computer system according to one embodiment;

Figure 3 illustrates an example of a motion control system having a PXI chassis including a computer card, motion control interface card, and measurement device according to one embodiment;

Figures 4A, 4C, and 4E illustrate example graphs of velocity versus time in a motion control system according to the prior art;

Figures 4B, 4D, and 4F illustrate example graphs of velocity versus time in a motion control system providing smoother operation according to various embodiments;

Figure 5 is a flowchart illustrating a motion control method using improved pulse placement for smoother operation according to one embodiment;

Figure 6 is a flowchart further illustrating the placement of pulses in a motion control method using improved pulse placement for smoother operation according to one embodiment; and

Figures 7A-B are flowcharts illustrating an algorithmic motion control method

using improved pulse placement for smoother operation according to one embodiment.

While the invention is susceptible to various modifications and alternative forms, specific embodiments thereof are shown by way of example in the drawings and will herein be described in detail. It should be understood, however, that the drawing and detailed description thereto are not intended to limit the invention to the particular form disclosed, but on the contrary, the intention is to cover all modifications, equivalents and alternatives falling within the spirit and scope of the present invention as defined by the appended claims.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Figure 1 – Exemplary Motion Control / Measurement System

Figure 1 illustrates an example motion control system 100 that includes various options for measurement or data acquisition. The motion control system may be configured to move object 150 and/or scan object 150 while object 150 is being moved. Figure 1 is exemplary only, and the present invention may be used in any of various systems, as desired.

The system 100 includes a host computer 102. The host computer 102 comprises a CPU, a display screen, memory, and one or more input devices such as a mouse or keyboard as shown. The computer 102 couples to a motion control system including a motion device 136 and a motion control interface card 138. As used herein, the term "motion device" or "motion control device" is intended to include stepper motors, servo motors, and other motion control devices or systems that are operable to receive a motion control signal and move responsive to the received signal. Typically an object is placed on or otherwise coupled to the motion device, and the motion device operates to move the object. The motion device 136 is coupled to the computer 102 through the motion control interface card 138. The motion control interface card 138 is typically plugged into an I/O slot in the computer 102, such as a PCI bus slot provided by the computer 102. However, the card 138 is shown external to computer 102 for illustrative purposes. The card 138 may also be implemented as an external device coupled to the computer 102. In one embodiment, a power drive or wiring interface 137 may convert control signals from the motion control interface card 138 into current and voltage power signals for the motion device 136.

The computer system 102 may also couple to one or more measurement devices which may be used, for example, to acquire measurements of an object 150 which is moved by the motion control device 136. The one or more measurement devices may include a GPIB instrument 112 and associated GPIB interface card 122, a data acquisition

(DAQ) board 114 and associated signal conditioning circuitry 124, a VXI/VME instrument 116, a PXI instrument 118, a video device 132 and associated image acquisition card 134, and/or one or more computer based instrument cards 142, among other types of measurement or data acquisition devices.

The GPIB instrument 112 is coupled to the computer 102 via a GPIB interface card 122 provided by the computer 102. In a similar manner, the video device 132 is coupled to the computer 102 via the image acquisition card 134. The data acquisition board 114 is coupled to the computer 102, and optionally interfaces through signal conditioning circuitry 124 to the UUT. The signal conditioning circuitry 124 preferably comprises a SCXI (Signal Conditioning eXtensions for Instrumentation) chassis comprising one or more SCXI modules 126.

As described above with respect to the motion control interface card 138, the GPIB card 122, the image acquisition card 134, and the DAQ card 114 are typically plugged in to an I/O slot in the computer 102, such as a PCI bus slot provided by the computer 102. However, these cards 122, 134 and 114 are shown external to computer 102 for illustrative purposes. The cards 122, 134 and 114 may also be implemented as external devices coupled to the computer 102, such as through a serial bus.

The VXI/VME chassis or instrument 116 is coupled to the computer 102 via a serial bus, MXI bus, or other serial or parallel bus provided by the computer 102. The computer 102 preferably includes VXI interface logic, such as a VXI, MXI or GPIB interface card (not shown), which interfaces to the VXI chassis 116. The PXI chassis or instrument is preferably coupled to the computer 102 through the computer's PCI bus.

A serial instrument (not shown) may also be coupled to the computer 102 through a serial port, such as an RS-232 port, USB (Universal Serial bus) or IEEE 1394 or 1394.2 bus, provided by the computer 102.

<u>Figure 2 – Exemplary Motion Control System for Scanning an Object</u>

Figure 2 illustrates an example motion control system of Figure 1, wherein the

system includes motion control interface device 138 and a data acquisition device 114. The motion control interface device 138 may be coupled to move a sensor 170 to scan an object. The sensor 170 may be operable to acquire measurements of the object 150 being scanned. The data acquisition device 114 may be coupled to the sensor 170 to acquire data or measurements from the sensor 170.

As shown, the motion control interface device 138 is directly coupled with the measurement device through a dedicated channel to provide real time triggering and/or communication between the motion control interface device 138 and the data acquisition device 114. The computer 102 may operate to receive and integrate or correlate the position data and measurements received from the motion control interface card 138 and data acquisition device 114, respectively, as described below.

Figure 2A illustrates an example motion control system wherein the motion control interface device 138 and data acquisition (or measurement) device 114 (not shown in Figure 2A) are comprised in computer system 102. The motion control interface device 138 controls motion control stage 136, which moves sensor 170 relative to the object 150 being scanned. The data acquisition device 114 is operable to acquire data sensed by the sensor 170.

Figure 2B illustrates an example motion control system wherein the motion control interface device 138 and image acquisition (or measurement) device 134 (not shown in Figure 2B) are comprised in computer system 102. The motion control interface device 138 controls motion control stage 136, which moves camera 132 relative to the object 150 being scanned. Here the camera 132 is simply one example of a sensor 170. The image acquisition device 134 is operable to acquire data sensed by the camera 132.

Figure 3 – Exemplary PXI-Based Motion Control System

Figure 3 illustrates an example motion control system of Figure 1, wherein the system includes a PXI chassis 118 comprising a computer card 102A, motion control interface card 138A and a measurement device, such as data acquisition device 114A. The

motion control interface card 138A is similar to the motion control interface card 138, except that the motion control interface card 138A is in a PXI card form factor. Similarly, the data acquisition device 114A is similar to the data acquisition device 114, except that the data acquisition device 114A is in a PXI card form factor.

As described above with respect to Figure 2, the motion control interface device 138A may be coupled to move a sensor 170 to scan an object. The sensor 170 may be operable to acquire measurements of the object 150 being scanned. The data acquisition device 114A may be coupled to the sensor 170 to acquire data or measurements from the sensor 170.

In this embodiment, the motion control interface device 138 is directly coupled with the measurement device through dedicated trigger and/or communication lines provided in the PXI backplane. Thus the PXI backplane provides real time triggering and/or communication between the motion control interface device 138A and the data acquisition device 114A. The computer or controller board 102A may be comprised in the PXI chassis to receive and integrate or correlate the position data and measurements received from the motion control interface card 138A and data acquisition device 114A, respectively, as described below.

Figure 4 – Examples Graphs Showing Smoother Operation in Various Embodiments

Figures 4A through 4F illustrate differences between motion control systems according to the prior art and according to embodiments of the present invention.

Figure 4A illustrates an example graph of velocity 402 versus time 404 at 700,000 steps per second in a motion control system according to the prior art. Figure 4B illustrates an example graph of velocity 402 versus time 404 at 700,000 steps per second in a motion control system providing smoother operation according to one embodiment.

Figure 4C illustrates an example graph of velocity 402 versus time 404 at 200,000 steps per second in a motion control system according to the prior art. Figure 4D illustrates an example graph of velocity 402 versus time 404 at 200,000 steps per second in a motion

control system providing smoother operation according to one embodiment.

Figure 4E illustrates an example graph of velocity 402 versus time 404 at 10,000 steps per second in a motion control system according to the prior art. Figure 4F illustrates an example graph of velocity 402 versus time 404 at 10,000 steps per second in a motion control system providing smoother operation according to one embodiment.

Each of the graphs may represent thousands of steps needed by the motion device to reach a desired position. Although the prior art implementations as illustrated by way of example in Figures 4A, 4C, and 4E typically do generate the correct number of steps at the correct average velocity, they do so at the expense of short-term error as shown in the "choppiness" of Figure 4A, 4C, and 4E.

The motion control system and method according to one embodiment may place the step pulses more accurately within the loop period. By using a delay time, the pulse train may be shifted to an arbitrary location within the loop period rather than evenly distributed throughout the loop period as in the prior art implementations. As shown in the following example using a loop period having a duration of 10 clocks and a step rate of 7 clocks, the use of delays according to one embodiment results in smoother motion than the "jerky" prior art method:

Step pulse	1	2	3	4	5	6	7	8
Prior art location	10	20	25	30	40	45	50	60
Delay location	7	14	21	28	35	42	49	56

The motion control system and method according to one embodiment may correct for quantization errors in the step generation due to clock limits in the digital system. For example, suppose the step rate in the above example was changed to 2.4 clocks/step per step with a loop period 10 clocks in duration:

Period	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Target Position	4.2	8.3	12.5	16.7	20.8	25	29.2
Steps to generate	4	4	4	4	4	5	4

Actual step rate	3	2	3	2	3	2	2
Clocks in period (including borrowed clocks)	12	8	12	8	12	10	8

If the digital motion control system can only output step rates that are an integer number of clocks, it can only choose a step rate of 2 or 3 clocks (rather than the ideal 2.4 clocks). If it uses a step rate of 2 clocks, then the pulse train will end early in the period and leave dead time that creates jumps in velocity. If it uses a step rate of 3 clocks, the pulse train will not finish by the end of the period and will run into the next period. In one embodiment, the solution is to generate a pulse train at the slower rate (3 clocks in this example) and correct for the "borrowed time" in the next loop iteration. Instead of assuming that each loop period is constant, according to one embodiment of the motion control system and method, the autocorrecting algorithm removes the borrowed time from its calculations and therefore allows the step generation to catch up at appropriate intervals. In the above example, the actual step rate will change between 2 and 3 clocks in order to achieve an average step rate of 2.4 clocks.

In one embodiment, a further improvement may be made to improve the accuracy of the motion control system and method. By allowing the step rate to change at a programmable point in the middle of the loop iteration (i.e., the series of loop periods), the step pulse generator may allow the steps generated to consume only the loop period and thus eliminate the borrowing of time from future loop periods. For example, revisit the first two iterations of the previous example (loop period: 10 clocks; step rate: 2.4 clocks; actual step rate: 2 or 3 clocks) to illustrate the method incorporating changeable step rates (as indicated in the "new location" row):

Step pulse	1	2	3	4	5	6	7	8
Borrowed location	3	6	9	12	14	16	18	20
New location	3	6	8	10	13	16	18	20

The borrowing method results in the 4 step pulses of the first iteration being spread out over 12 clocks at a rate of 3 clocks. On the second iteration it generates the step pulses over 8 clocks at a rate of 2 clocks. The method using a changeable loop period results in the 4 step pulses of the first iteration being spread out over 10 clocks at a rate of 3 clocks for the first two steps and 2 clocks for the last two steps. On the second iteration it generates the same step pattern. This method removes the need for the autocorrection of borrowed time and allows for an even more accurate pulse placement within the loop period.

Figures 5-7 – Flowcharts of the Motion Control Method Featuring Smoother Operation

Figures 5, 6, 7A, and 7B are flowcharts further illustrating the motion control method discussed above with reference to Figure 4B. As discussed with reference to Figures 1 through 4, a motion device is operable to move an object. In one embodiment, the motion device includes a stepper motor. A motion control system is coupled to the motion device. The motion control system may include a computer system and a motion controller. The motion control system may include a processor and a memory medium, wherein the memory medium stores a motion control software program which is executable by the processor. A power drive may be coupled to the motion device and the motion control system. The power drive may be operable to receive the pulses from the motion controller, translate the pulses into power signals, and send the power signals to the motion device.

Figure 5 is a flowchart illustrating a method for controlling motion of an object according to one embodiment. In 601, a placement of pulses may be determined for each of a plurality of time intervals such that the pulses are placed evenly across the plurality of time intervals, wherein the quantity of pulses in each of the time intervals is variable. In 603, the pulses may be generated across the time intervals according to the placement determined in 601. In 605, the pulses may drive the motion device to move the object.

In one embodiment, in determining the placement of pulses for each of the plurality

of time intervals, a delay may be used to place each pulse at an arbitrary location within one of the time intervals. The time intervals may be variable or fixed in length in various embodiments. Where the time intervals are fixed in length, a pulse rate may be changed within one of the time intervals.

Figure 6 is a flowchart further illustrating step 601 from Figure 6a according to one embodiment. In 611, a placement of pulses may be determined for a first time interval at a first rate of pulse generation per time interval. The first rate may have a value of 1 plus an integer portion of a desired fractional rate of pulse generation per time interval. In 613, a placement of pulses may be determined for a second time interval following the first time interval at a second rate having a value of the integer portion of the desired fractional rate of pulse generation.

At both high and low velocities, the method illustrated in Figures 5 and 6 will typically result in smoother operation as well as achieve positional accuracy through accurate pulse placement.

Figures 7A and 7B are flowcharts further illustrating a detailed algorithm which implements the method shown Figures 5 and 6 according to one embodiment. Although the algorithm of Figures 7A and 7B assumes that the motion controller 138 includes a field-programmable gate array (FPGA), the method may be implemented using any suitable motion controller. An FPGA is a semi-conductor device that contains a large quantity of gates (logic devices) which are not interconnected and whose function is determined by a wiring list which is downloaded to the FPGA. The wiring list determines how the gates are interconnected, and this interconnection is performed dynamically by turning semiconductor switches on or off to enable the different connections.

In 701, perform initial calculations such as maxClocks = defaultPIDclocks[PIDRate] – maxClocksOvershoot. The value of wholeSteps may also be generated in 701. The variable maxClocks may store the number of FPGA clock cycles that may be consumed in a given time slice (i.e., loop period). This value may start as

defaultPIDClocks for each time slice, but it may be adjusted to account for any borrowed time from the previous time slice. The variable defaultPIDClocks relates to the set number of FPGA clock cycles that can be consumed for each PID rate a user can select (where different PID rates change the size of a time slice). The variable wholeSteps represents the number of step pulses that need to be outputted in a given time slice.

In 703, determine whether wholeSteps = 0. If wholeSteps = 0, then in 739, set parameter values and proceed to 741. If wholeSteps is nonzero, then in 705, assign Period = maxClocks/Velocity. Period is the actual number of FPGA clocks a step pulse should take; in one embodiment, this number may include fractional information.

In 707, assign deadband = fractional part of Position * Period.

In 709, calculate FPGAPeriod as an integer number representing the actual number of FPGA clocks a step pulse will take. In one embodiment, the FPGA cannot use fractional clocks.

In 711, assign pulseWidth = FPGAPeriod/4. The variable pulseWidth stores a step pulse's active time in number of FPGA clocks. In 713, determine whether pulseWidth > 255. If pulseWidth exceeds 255, then in 715, assign pulseWidth = 255. In 717, determine whether pulseWidth < 2. If pulseWidth is less than 2, then in 719, assign pulseWidth = 2.

In 721, assign delay = maxClocks - ((Period * (wholeSteps - 1)) + deadband). The variable delay is the number of FPGA clocks before the first step pulse is to be generated in a time slice. The variable deadband is a calculated value representing the time between the start of the last step pulse to be generated and the end of the time slice. This value is used to correctly position step pulses with in a time slice.

In 723, add the delay/Adjust to delay and maxClocks.

In 725, determine whether wholeSteps > 1. If wholeSteps is greater than one, then in 727, calculate maxClocksOvershoot. The variable maxClocksOverShoot stores the number of FPGA clocks that are borrowed from the next time slice.

In 729, determine whether FPGAPeriod <= maxClocks. If true, then in 731, fix the

deadbandDelayPeriod between time slices. The variable deadbandDelayPeriod may be formed by adding the previous time slice's deadband and the current time slice's delay. This is the period of a step pulse that crosses over the time slice boundary.

In 733, determine whether delay < pulseWidthOverlap. If true, then in 735, adjust delay for pulseWidth. The variable pulseWidthOverlap is the number of FPGA clock cycles that the previous time slice's last step pulse's pulseWidth overlaps into the current time slice.

In 737, calculate pulseWidthOverlap for the next time slice. In 741, store deadband and FPGAPeriod for use in next time slice. In 743, write values to the FPGA. The method may proceed again for a next time slice.

Various embodiments may further include receiving or storing instructions and/or data implemented in accordance with the foregoing description upon a carrier medium. Suitable carrier media may include storage media or memory media such as magnetic or optical media, e.g., disk or CD-ROM, as well as transmission media or signals such as electrical, electromagnetic, or digital signals, conveyed via a communication medium such as network and/or a wireless link.

While the present invention has been described with reference to particular embodiments, it will be understood that the embodiments are illustrated and that the invention scope is not so limited. Any variations, modifications, additions and improvements to the embodiments described are possible. These variations, modifications, additions and improvements may fall within the scope of the invention as detailed within the following claims.